Project 4 User Manual

1. Program Introduction:

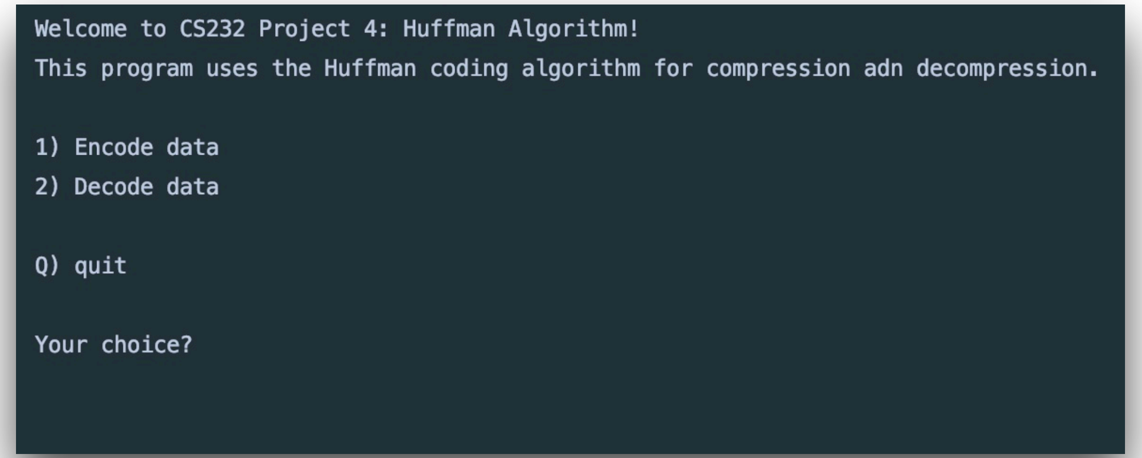
This menu driven program uses the Huffman coding algorithm for compression. Any file can be compressed by this method, often with substantial savings. Decompression will faithfully reproduce the original. You will be given options to choose encode or decode a text file.

2. Input Requirements and Restrictions:

- For Main menu options, inputs are integers: **1, 2 or Q.**
- For file name input, inputs are string type file name including its path.

3. Executing the Program:

Once the program executes, it will display greeting messages and main menu options, inputs are either 1 or 2, if want to terminate the program, enter q or Q.

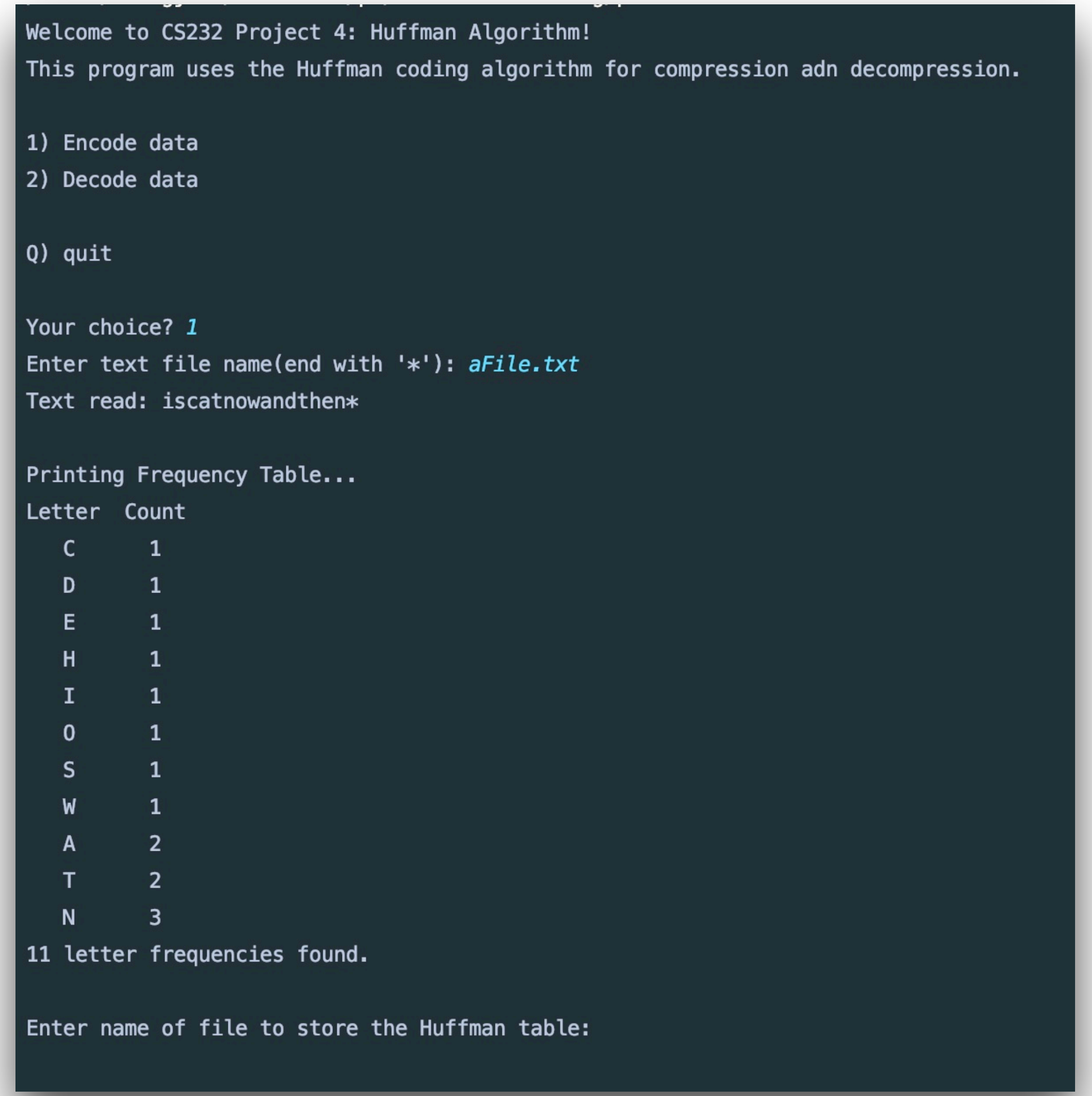


```
Welcome to CS232 Project 4: Huffman Algorithm!
This program uses the Huffman coding algorithm for compression adn decompression.

1) Encode data
2) Decode data

Q) quit

Your choice?
```

- **Enter 1:** encode data from a file and store the Huffman Table and encode to files.
- **Enter 2:** decode data from files containing Huffman Table and encode.
- **Enter Q or q:** terminate the program.
- **Other inputs:** are considered invalid, user will be given unlimited chances to re-enter.

4. Encode Data:

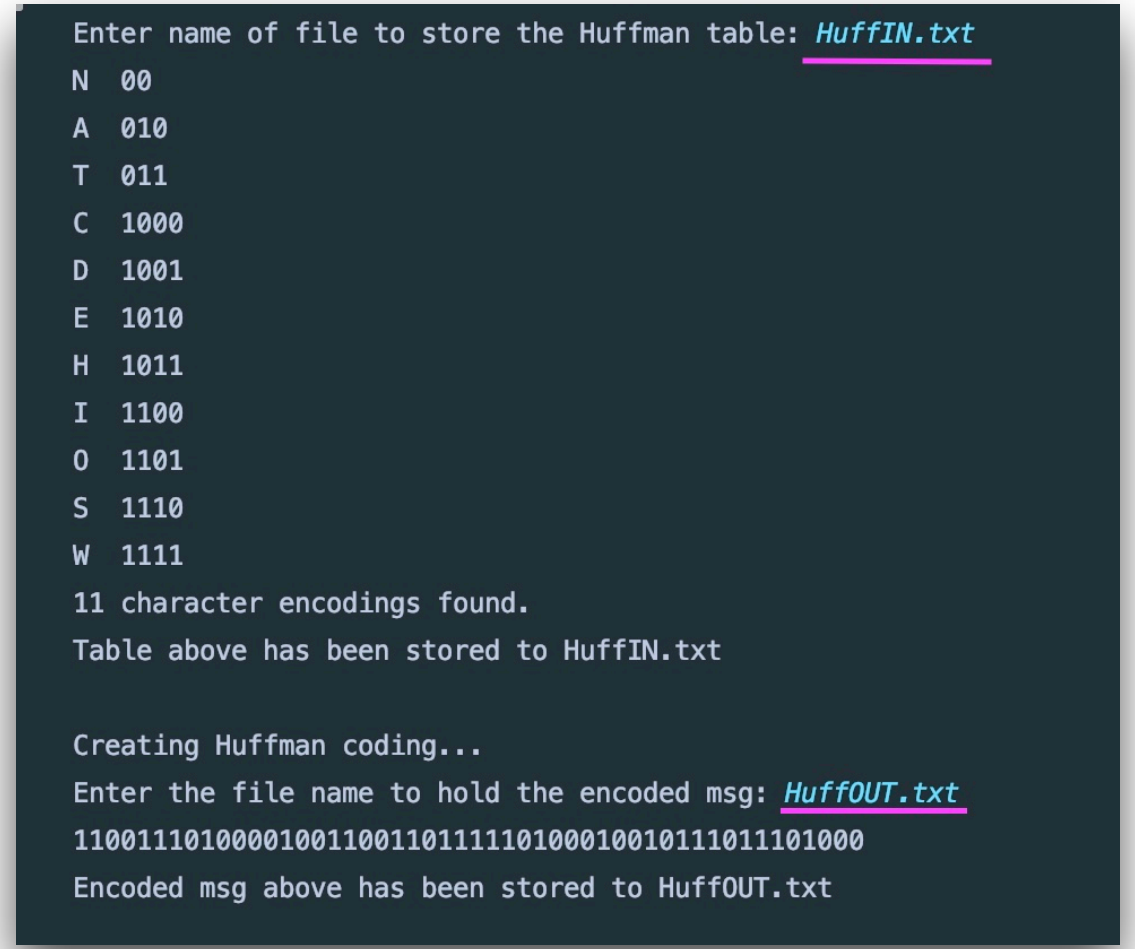
Assuming the file to be encoded is in the same directory of the executable program named **aFile.txt**. If the file is loaded successfully, console will display text data just read from file and frequency count table for user to verify:



```
Welcome to CS232 Project 4: Huffman Algorithm!
This program uses the Huffman coding algorithm for compression adn decompression.

1) Encode data
2) Decode data

Q) quit

Your choice? 1
Enter text file name(end with '*'): aFile.txt
Text read: iscatnowandthen*

Printing Frequency Table...
Letter  Count
   C      1
   D      1
   E      1
   H      1
   I      1
   O      1
   S      1
   W      1
   A      2
   T      2
   N      3
11 letter frequencies found.

Enter name of file to store the Huffman table:
```

- **Output Huffman Table and Encoding:**

Now user can enter file name with .txt extension to store Huffman Table and encode. Example below uses ***HuffIN.txt*** and ***HuffOUT.txt*** for **Huffman Table** and **encode** respectively:

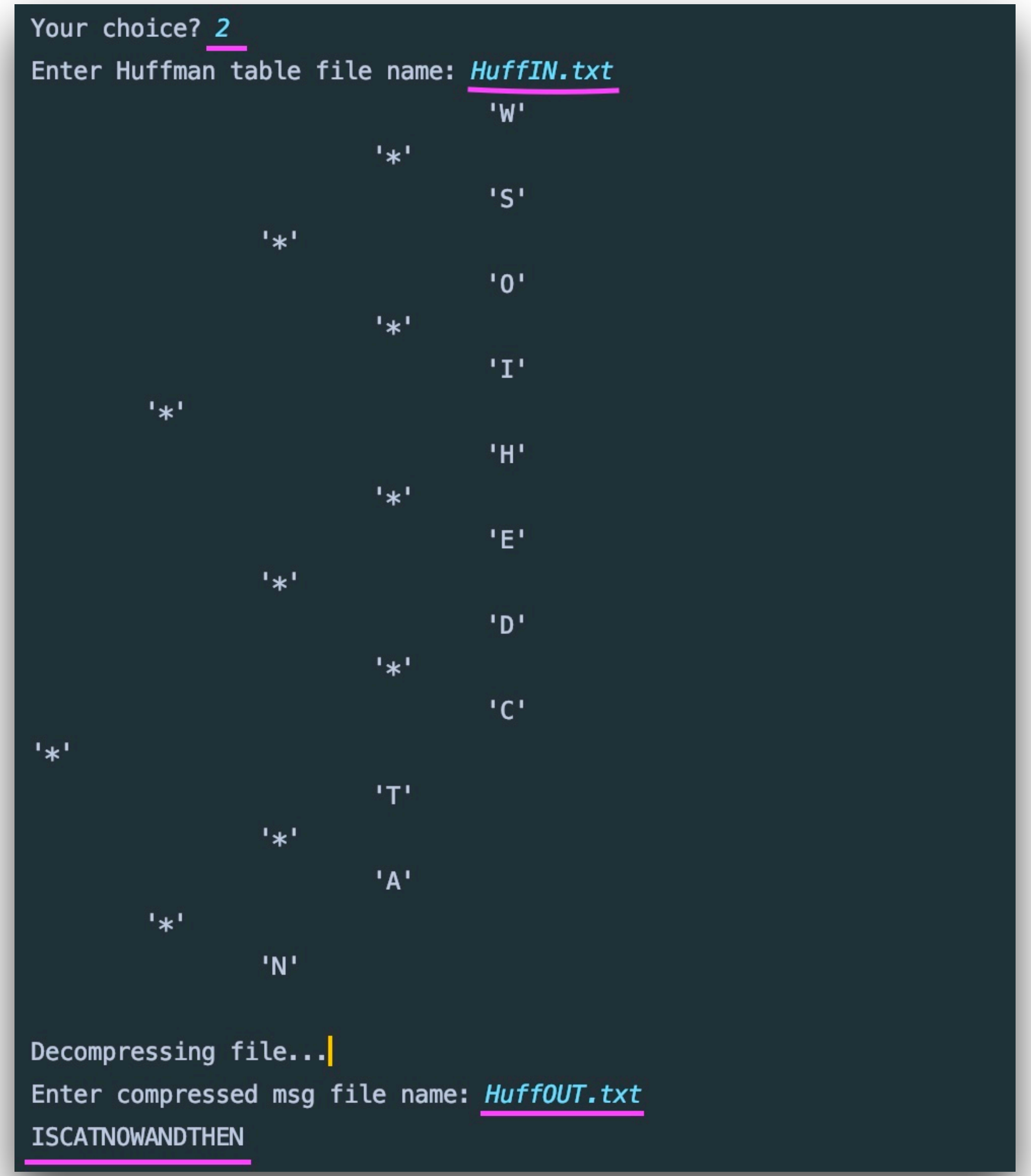


```
Enter name of file to store the Huffman table: HuffIN.txt
N  00
A  010
T  011
C  1000
D  1001
E  1010
H  1011
I  1100
O  1101
S  1110
W  1111
11 character encodings found.
Table above has been stored to HuffIN.txt

Creating Huffman coding...
Enter the file name to hold the encoded msg: HuffOUT.txt
11001110100001001100110111110100010010111011101000
Encoded msg above has been stored to HuffOUT.txt
```

5. Decode Data:

User will be prompted for Huffman Table file and compressed msg file. Using the example above, we can use ***HuffIN.txt*** to decode ***HuffOUT.txt***.

- If ***HuffIN.txt*** file is loaded successfully, program will construct Huffman Tree from Huffman table data in the file and print the tree to console.
- If ***HuffOUT.txt*** is loaded successfully, console will print the decoded msg to console from encoding tree just constructed.



```
Your choice? 2
Enter Huffman table file name: HuffIN.txt
                                'W'
                        '*'
                                'S'
                '*'
                                'O'
                        '*'
                                'I'
        '*'
                                'H'
                        '*'
                                'E'
                '*'
                                'D'
                        '*'
                                'C'
'*'
                        'T'
                '*'
                        'A'
        '*'
                'N'

Decompressing file...
Enter compressed msg file name: HuffOUT.txt
ISCATNOWANDTHEN
```